NETWORK COMPUTER SYSTEM AND SUBSTITUTE COMPILING SERVER DEVICE

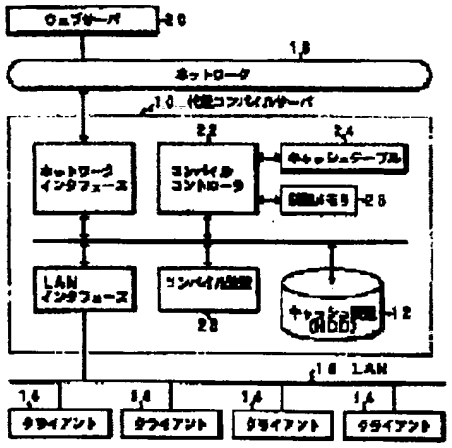
Patent number: JP2000215181
Publication date: 2000-08-04
Inventor: SHIMURA HIROYA
Applicant: FUJITSU LTD
Classification:
- international: G06F15/16; G06F9/45; G06F9/46
- european:
Application number: JP19990013011 19990121
Priority number(s):

Also published as:
US6370687 (B1)

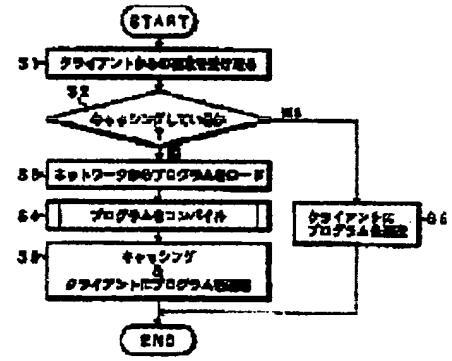
Abstract of **JP2000215181**

PROBLEM TO BE SOLVED: To enable a client to fast execute a virtual machine computer program in a network by giving a compiling function to a substitute server.
SOLUTION: A client 14 is prepared to execute a virtual machine computer program in a network 18 together with a substitute compiling server 10 which receives the virtual machine computer program from the network 18 and transfers it to the client 14, i.e., a requester after compiling the program. Thus, a compiling function of the virtual machine computer program is given to a substitute server in the network 18 to obtain the server 10. As a result, the burden of the client 14 which executes the virtual machine computer program is reduced and the client 14 can fast execute an application program in a network computer system.

(A)



(B)



(19)日本国特許庁（ＪＰ）　(12) **公開特許公報**（Ａ）　(11)特許出願公開番号
特開2000－215181
（P2000－215181A）
(43)公開日　平成12年８月４日(2000.8.4)

(51)Int.Cl.'		識別記号	FI		テーマコード'(参考)
Ｇ０６Ｆ	15/16	６３０	Ｃ０６Ｆ 15/16	６３０Ｂ	５Ｂ０４５
		６２０		６２０Ｂ	５Ｂ０８１
	9/45		9/46	３６０Ｂ	５Ｂ０９８
	9/46	３６０	9/44	３２２Ａ	

審査請求　未請求　請求項の数12　ＯＬ　（全 16 頁）

(21)出願番号　特願平11－13011

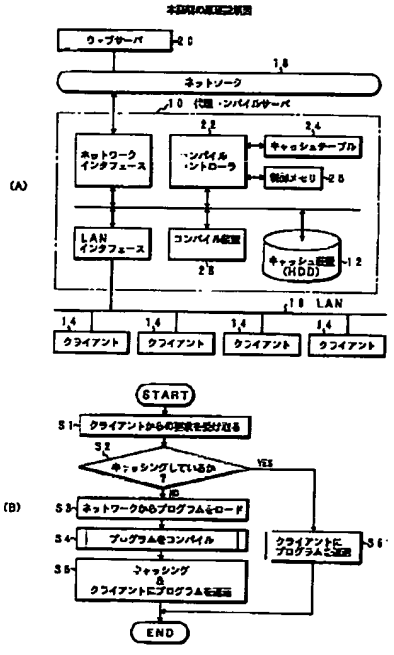
(22)出願日　平成11年１月21日(1999.1.21)

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(72)発明者　志村　浩也
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内
(74)代理人　100079359
弁理士　竹内　進　（外１名）

Ｆターム(参考)　5B045 AA00 BB47 DD12 GC09
5B081 CC41
5B098 AA10 GC16 HH01

(54)【発明の名称】　ネットワークコンピュータシステム及び代理コンパイルサーバ装置

(57)【要約】
【課題】代理サーバにコンパイル機能を持たせ、クライアントでネットワーク上の仮想マシンコンピュータプログラムの高速実行を可能とする。
【解決手段】ネットワーク１８上の仮想マシンコンピュータプログラムを実行するクライアント１４と、クライアント１４の要求に基づき、ネットワーク１８上から仮想マシンコンピュータプログラムを受け取り、コンパイルして要求元のクライアントに受け渡す代理コンパイルサーバ１０とを備える。



とのできる等の理由により急速に広まっている。

【０００３】このようなＪａｖａ言語で記述されたＪａｖａプログラムは、プラットフォームを選ばないＪａｖａ仮想マシン（ＪＶＭ；Java Virtual Machine）のプログラムであるバイトコードとして配布されるため、クライアントの計算機では直接実行することができず、Ｊａｖａプログラムの実行は、クライアント側でＪａｖａ仮想マシンのコードをエミュレートする必要があり、インタプリタ方式で実行される。

【０００４】インタプリタ方式は、ソフトウェアによりバイトコードを逐次解釈して実行するエミュレーションであり、他の言語と比較して実行が遅くなる。これを解決する方法としてＪＩＴコンパイラ(Just In Time Compiler) をクライアント側で使う手法がある。ＪＩＴコンパイラは、バイトコードのプログラムをネットワークからロードして実行する前に、バイトコードをクライアントマシンのネイティブコードにコンパイルすることにより、直接、クライアントのハードウェアで実行でき、高速な実行が可能となる。

【０００５】図１に従来のウェブサーバ（Web Server）のアクセスにプロキシ・サーバ（Proxy Server）を使ったシステムの構成を示す。プロキシサーバを使った場合の基本的な処理の流れは次のようになる。ＬＡＮ１０２に接続されたクライアント１００－１～１００－４は、通常、ｈｔｔｐプロトコルを用いてプロキシサーバ１０４にウェブサーバ１１０のページ、イメージ、プログラム等を取ってくるように要求を出す。

【０００６】例えばクライアント１００－２が要求を出すと、プロキシサーバ１０４は要求されたデータがハードディスク等のキャッシュ１０６上に保存してあるかどうか調べ、保存されていない場合、外部のネットワーク１０８を介してウェブサーバ１１０にアクセスし、必要なデータを取ってくる。

【０００７】プロキシサーバ１０４は、ハードディスク等のキャッシュ上１０６にウェブサーバ１１０から獲得したデータを保存すると同時に、要求元のクライアント１００－２にデータを返す。もし、データがキャッシュ１０６上に保存されていた場合は、キャッシュ１０６からデータを取り出し、クライアント１００－２に返送する。

【０００８】このため、従来、ネットワーク上のウェブサーバ１１０に存在するＪａｖａプログラムを例えばクライアント１００－２が実行するときは、プロキシサーバ１０６からＪａｖａプログラムを受け取ったクライアント１００－２が、ＪａｖａプログラムをＪＩＴコンパイラでコンパイルしながら実行している。

【０００９】

【発明が解決しようとする課題】しかしながら、従来のクライアント側にＪＩＴコンパイラを設けた場合は、クライアントでＪａｖａプログラムのバイトコードをコンパイルするための時間が必要となり、プログラムの実行に影響を与える。特に、サーバからクライアントという一方向のデータの流れではなく、クライアントからの要求をサーバに戻して処理したり、クライアントで処理した結果をサーバに渡す等の双方向のデータの流れをもつインタラクティブな環境では、プログラムの立ち上がり時間やレスポンスを速くすることが重要であり、コンパイルに時間をかけることはできない。

【００１０】更に、プログラムの実行を高速化するためには、ＪＩＴコンパイラでコンパイルを行うときに複雑な最適化を行う必要があるが、複雑な最適化を行うほどコンパイルの時間が遅くなる。また最適化を行わないと、コンパイル時間は速くなるが、実行時間は長くなるというトレードオフがある。

【００１１】本発明は、代理サーバにコンパイル機能を持たせることで、クライアントでネットワーク上の仮想マシンコンピュータプログラムの高速実行を可能とするコンピュータシステム及び代理コンパイルサーバ装置を提供することを目的とする。

【００１２】

【課題を解決するための手段】本発明は、図１（Ａ）のように、ネットワーク１８上の仮想マシンコンピュータプログラムを実行するクライアント１４と、クライアント１４の要求に基づき、ネットワーク１８上のウェブサーバ２０から仮想マシンコンピュータプログラム、例えばＪａｖａプログラムを受け取り、コンパイルして要求元のクライアントに引き渡す代理コンパイルサーバ１０とを備えたことを特徴とするネットワークコンピュータシステムである。

【００１３】このように本発明は、高速な代理コンパイルサーバを用意し、代理コンパイルサーバにクライアント側でのＪＩＴコンパイルを肩代わりさせることにより、クライアントでのプログラムの実行を高速化させる。この場合、代理コンパイルサーバが十分高速であれば、コンパイル時に複雑な最適化を行うことができ、クライアントの実行速度が更に向上する。

【００１４】代理コンパイルサーバ１０は、クライアント１４とプロセッサ等の相違で実行形式が異なる場合、クライアント１４の実行形式に従って仮想マシンコンピュータプログラムをコンパイルする。例えば代理コンパイルサーバ１０は、クライアント１４の実行形式を予め把握しており、要求元のクライアント１４の実行形式に従って仮想マシンコンピュータプログラムをコンパイルする。また代理コンパイルサーバ１０は、クライアント１４が要求した実行形式に従って仮想マシンコンピュータプログラムをコンパイルしてもよい。

【００１５】更に、代理コンパイルサーバ１０は、ネットワーク上のウェブサーバ２０から仮想マシンコンピュータプログラムを受け取った際に、接続可能なクライアント１４の実行形式の種別毎にコンパイルし、複数の実

行形式のコンパイル済みプログラムをクライアント１４に受け渡し、要求元のクライアント１４に自己の実行形式に合うコンパイル済みプログラムを選択させて実行させてもよい。

【００１６】代理コンパイルサーバ１０は、図１（Ｂ）のように、コンパイル済みのプログラムを要求元のクライアント１４に引き渡すと同時にキャッシュ１２上に保存しておき、クライアント１４から要求があった時に、キャッシュ１２上に保存しているコンパイル済みプログラムを引き渡す。

【００１７】このため、代理コンパイルサーバ１０が高速でなかったとしても、コンパイルしたプログラムをキャッシュ１２上に保存しておくことにより、最初の実行はコンパイルに時間がかかり実行が遅くなるが、２度目からの実行は高速に実行可能となる。

【００１８】代理コンパイルサーバ１０は、クライアント１４から要求があった際に、次にクライアント１４が要求する仮想マシンコンピュータプログラムを予測してネットワーク上のウェブサーバ２０から受け取り、コンパイルした後にキャッシュ１２上に保存する。このような予測コンパイルによって、クライアント１４の実行を更に高速化できる。

【００１９】代理コンパイルサーバ１０は、ネットワーク１８から受け取った仮想マシンコンピュータプログラムをキャッシュ１２上に保存しておき、クライアント１４から要求があった時に、キャッシュ１２上に保存している仮想マシンコンピュータプログラムをコンパイルしてクライアント１４に受け渡す。これによってキヤッシュ１２の記憶容量を低減する。

【００２０】この場合にも、代理コンパイルサーバ１０は、クライアント１４から要求があった際に、要求された仮想マシンコンピュータプログラムに加えて次にクライアント１４が要求する仮想マシンコンピュータプログラムを予測してネットワーク１８上のウェブサーバ２０から受け取り、キャッシュ１２上に保存する。

【００２１】代理コンパイルサーバ１０は、ネットワーク上のウェブサーバ２０から仮想マシンコンピュータプログラムとして受け取ったバイトコードを、要求元のクライアント１４の実行形式に適合したマシンコード（ネイティブコード）にコンパイルして引き渡す。

【００２２】また本発明は、代理コンパイルサーバ１０を提供することを特徴とし、代理コンパイルサーバ１０は、ネットワーク１８上の仮想マシンコンピュータプログラムを実行するクライアント１４の要求に基づき、ネットワーク１８上から要求された仮想マシンコンピュータプログラムを受け取るコントローラ２２と、コントローラ２２で受け取ったプログラムをコンパイルして要求元のクライアントに受け渡すコンパイル装置２８とを備える。

【００２３】代理コンパイルサーバ装置１０は、コンパイル装置２８でコンパイルしたプログラムを保存しておき、クライアント１４から要求があった時に、コンパイル済みプログラムを受け渡すキャッシュ１２を設ける。

【００２４】

【発明の実施の形態】図２は、本発明のネットワークコンピュータシステムのシステム構成図である。図２において、ＬＡＮ１６に接続されたクライアント１４－１，１４－２，１４－３，１４－４に対しては、インターネット等のネットワーク１８のウェブサーバ２０とのアクセスのため代理コンパイルサーバ１０が設けられている。

【００２５】代理コンパイルサーバ１０としては、図１６と同様、プロキシサーバが使用され、このプロキシサーバにクライアント１４－１～１４－４がウェブサーバ２０に要求した仮想マシンコンピュータプログラムであるＪａｖａプログラムのコンパイル機能を付加することで、代理コンパイルサーバ１０を実現している。代理コンパイルサーバ１０にはハードディスク等を用いたキャッシュ１２が接続されている。

【００２６】図３は、図２のネットワークコンピュータシステムに設けた本発明による代理コンパイルサーバ１０の機能ブロック図であり、ウェブサーバ及びクライアント側と共に表わしている。

【００２７】プロキシサーバを用いた代理コンパイルサーバ１０には、コンパイルコントローラ２２、キャッシュテーブル２４、制御メモリ２６、コンパイル装置２８、ＬＡＮインタフェース３０及びネットワークインタフェース３２が設けられる。

【００２８】ここでネットワーク１８上のウェブサーバ２０には、仮想マシンコンピュータプログラムとしてのＪａｖａプログラムがウェブページ上のアプレットとして準備されており、クライアント１４－１～１４－４側にはホットＪａｖａブラウザが準備され、ウェブサーバ２０に対するアクセスで受け出された仮想マシンコンピュータプログラムであるＪａｖａプログラムのバイトコードをエミュレートしてインタプリタ方式で実行する環境が設けられている。

【００２９】代理コンパイルサーバ１０のコンパイルコントローラ２２は、クライアント１４－１～１４－４からｈｔｔｐプロトコルを用いたＪａｖａプログラムを取得するための要求を受けると、キャッシュテーブル２４を参照し、キャッシュ装置１２に要求されたＪａｖａプログラムのコンパイルプログラムが保存されているか否かチェックし、保存されていなければネットワーク１８を経由してウェブサーバ２０にＪａｖａプログラムの引き渡しを要求する。

【００３０】ウェブサーバ２０から代理コンパイルサーバ１０にＪａｖａプログラムが受け渡されると、コンパイルコントローラ２２はコンパイル装置２８を制御し、ウェブサーバ２０から受け渡されたＪａｖａプログラム

のバイトコードを、そのとき要求を行っているクライアントの実行環境に適合したネイティブ環境のマシンコードにコンパイルし、コンパイル済みのＪａｖａプログラムを要求元のクライアントに引き渡す。同時にキャッシュ装置１２にコンパイル済みのＪａｖａプログラムを保存し、キャッシュテーブル２４にキャッシュ結果を登録する。

【００３１】図4は、図3の代理コンパイルサーバ１０に設けたキャッシュ装置１２の一例であり、この例ではウェブサーバ２０から引き渡されたコンパイル前のＪａｖａプログラム３４－１，３４－２と、それぞれをコンパイル装置２８でネイティブコードにコンパイルしたコンパイル済みＪａｖａプログラム３６－１，３６－２を保存している。

【００３２】また図3の代理コンパイルサーバ１０のコンパイルコントローラ２２に対して設けられた制御メモリ２６には、例えば図5のように、代理コンパイルサーバ１０にＬＡＮ１６を介してアクセス可能なクライアント１４－１～１４－４のサーバ名と、ネットワークのＪａｖａプログラムを実行するプロセッサ等で決まる実行形式の対応関係を登録している。

【００３３】制御メモリ２６のクライアント名Ａ，Ｂ，Ｃ，Ｄは、例えばネットワークアドレス「ｈｔｔｐ：／／ＷＷＷ．Ａ．ｃｏ．ｊｐ／」等が使用され、実行形式α，β，γとしては、クライアント１４－１～１４－４のプロセッサ名、例えばＳＰＡＲＣＶ８，Ｘ８６，Ｉｎｔｅｌ４８６等の識別子が設定される。

【００３４】この制御メモリ２６により代理コンパイルサーバ１０のコンパイルコントローラ２２は、クライアント側からＪａｖａプログラムの引き渡し要求を受けた際に、クライアントのサーバ名による制御メモリ２６の参照でクライアントの実行形式を認識し、コンパイル装置２８でＪａｖａプログラムのバイトコードからコンパイルする実行環境のネイティブコードを認識する。

【００３５】このようなクライアント側の実行形式に対応して、コンパイル装置２８には図6に示すように、クライアント側の実行形式α，β，γに対応したＪＩＴコンパイラ（ＪＩＴコンパイルプログラム）３８－１，３８－２，３８－３が予め準備されており、要求のあったサーバ名に対応した実行形式のＪＩＴコンパイラを選択してＪａｖａプログラムのバイトコードから対応する実行形式のネイティブコードへのコンパイルを行う。

【００３６】例えばＪＩＴコンパイラ３８－１はＪａｖａバイトコードをＳＰＡＲＣＶ８のネイティブコードに翻訳するコンパイラであり、ＪＩＴコンパイラ３８－２はＪａｖａバイトコードをＸ８６のネイティブコードに翻訳するコンパイラであり、更にＪＩＴコンパイラ３８－３はＪａｖａバイトコードをＩｎｔｅｌ４８６のネイティブコードに翻訳するコンパイラである。

【００３７】勿論、ＪＩＴコンパイラ３８－１～３８－３に加え、Ｊａｖａ仮想マシンコンピュータプログラムの実行環境を提供するホットＪａｖａブラウザがインストールされているクライアントマシンの種別に応じ、必要に応じて適宜にＪＩＴコンパイラを増設することができる。

【００３８】図7は、図3の代理コンパイルサーバ１０の処理動作のフローチャートである。いまステップＳ１で例えばクライアント１４－１からウェブサーバ２０に対するＪａｖａプログラムのアクセス要求を受け取ると、ステップＳ２で、要求されたＪａｖａプログラムがキャッシュ装置１２に保存されているか否かチェックする。

【００３９】具体的には、クライアント１４－１からのＪａｖａプログラムのアクセス要求に対しコンパイルコントローラ２２が、要求を行ったクライアント名により制御メモリ２６を参照して、その実行形式を認識し、続いてキャッシュテーブル２４を参照し、クライアントの実行形式に適合したコンパイル済みのＪａｖａプログラムがキャッシュ装置１２上に存在するか否かチェックする。

【００４０】キャッシュ装置１２上に存在しなければ、コンパイルコントローラ２２はネットワークインタフェース３２を介してネットワーク１８上のウェブサーバ２０に対し、クライアント１４－１から受けたプログラム要求を通知する。このプログラム要求を受けてウェブサーバ２０は、該当するＪａｖａプログラム即ちアプレットを代理コンパイルサーバ１０に返送する。

【００４１】ウェブサーバ２０から要求したＪａｖａプログラムを受け取った代理コンパイルサーバ１０のコンパイルコントローラ２２は、コンパイル装置２８にＪａｖａプログラムを引き渡すと共に、制御メモリ２６の参照で認識した要求元のクライアント１４－１のクライアント名に対応する実行形式に対応した図6のような複数のＪＩＴコンパイラのいずれかを指定して、ステップＳ４でＪａｖａプログラムのコンパイルを行う。

【００４２】コンパイル装置２８により要求元クライアント１４－１の実行形式のネイティブコードにメソッド単位に翻訳され且つ最適化されたコンパイル済みＪａｖａプログラムは、キャッシュ装置１２にキャッシングすると同時に、ＬＡＮインタフェース３０を介して要求元のクライアント１４－１に返送され、クライアント１４－１においてコンパイル済みのネイティブコードからなるＪａｖａプログラムが実行される。

【００４３】キャッシュ装置１２に対するＪａｖａプログラムのキャッシングは、図4に示したように、ウェブサーバ２０から受け取ったコンパイル前のＪａｖａプログラムとコンパイル装置２８でコンパイルしたコンパイル済みのＪａｖａプログラムの両方を保存する。

【００４４】このうちコンパイル済みのＪａｖａプログラムは要求元のクライアントの実行形式に対応したネイ

ティブコードのプログラムであり、キャッシングした後は、同じ実行形式のクライアントからの要求に対し応答する。

【００４５】これに対し実行形式の異なるクライアントから同じＪａｖａプログラムの要求があった場合には、キャッシュ装置１２に保存しているコンパイル前のＪａｖａプログラムを読み出し、そのときの要求元のクライアントの実行形式に対応したＪＩＴコンパイラを選択してコンパイルし、コンパイルしたＪａｖａプログラムは要求元に返送し、同時にキャッシュ装置１２に別の実行形式であるネイティブコードに翻訳されたＪａｖａプログラムを保存する。

【００４６】一方、図7のステップＳ２でクライアント１４－１から要求されたＪａｖａプログラムがキャッシュ装置１２にキャッシングされている場合には、ステップＳ６に進み、キャッシュ装置１２に保存していたコンパイル済みのＪａｖａプログラムを読み出してクライアント１４－１に返送する。

【００４７】このようにクライアント側からの要求したＪａｖａプログラムがキャッシュ装置１４上に保存されていてキャッシュヒットとなった際には、ウェブサーバ２０に対するＪａｖａプログラムの要求及び引き渡されたＪａｖａプログラムのコンパイルが不要となり、代理コンパイルサーバ１０よりコンパイル済みのＪａｖａプログラムを直ちに返送して、要求元のクライアントで実行することができる。

【００４８】このような本発明の代理コンパイルサーバ１０により、ＪＩＴコンパイラの機能がサーバ側で実現され、クライアントからネットワーク上のウェブサーバに対するＪａｖａプログラムの要求に対し、代理コンパイルサーバ１０より要求元のクライアントの実行形式に適合したネイティブコードにコンパイルされ且つ最適化されたＪａｖａプログラムが返送され、代理コンパイルサーバ１０としてプロキシサーバのような高性能のマシンを使用することで、Ｊａｖａプログラムのコンパイル及び最適化の処理時間がごく短時間で済み、結果としてクライアント１４－１がＪａｖａプログラムを要求してから実行するまでの処理時間を大幅に短縮し、Ｊａｖａプログラムをインタプリタ方式で実行した場合に比べ、より高速実行ができる。

【００４９】また代理コンパイルサーバ１０でＪａｖａプログラムをコンパイルして要求元のクライアントに返送するため、クライアントが小規模のマシンでＪＩＴコンパイラをもつことができない場合でも、ＪＩＴコンパイラをもつことのできるマシンと同様かそれ以上の実行スピードを得ることができる。

【００５０】更に代理コンパイルサーバ１０は、コンパイル済みのＪａｖａプログラムをキャッシュ装置にキャッシングしておくことで、２回目以降の同じＪａｖａプログラムの要求に対してはキャッシュ装置から読み出して返送することで直ちにクライアント側で実行でき、特にウェブサーバとクライアントとの間でプログラムやデータをやり取りするインタラクティブな動作環境において、コンパイルに要する時間を最小限にして、プログラムの立ち上がり時間やレスポンスを速くすることができる。

【００５１】図8は、図7のステップＳ４における図3のコンパイル装置２８のＪａｖａプログラムのコンパイル処理の詳細である。図8のコンパイル処理にあっては、図5に示した制御メモリ２６を利用し、要求のあったクライアント名から、その実行形式を認識する。

【００５２】このときコンパイル装置２８には、図6のようにクライアントの実行形式に適合した複数のＪＩＴコンパイラ３８－１，３８－２，３８－３が準備されていることから、ステップＳ１で認識した実行形式に適合したＪＩＴコンパイラを選択してＪａｖａプログラムのコンパイルを実行する。

【００５３】図9は、図7のステップＳ４におけるＪａｖａプログラムのコンパイルの他の実施形態のフローチャートである。このコンパイル処理にあっては、ステップＳ１でクライアントからのプログラムの要求メッセージを解析し、クライアントの実行形式を認識する。この場合には図5のような制御メモリ２６は不要となる。次にステップＳ２に進み、ステップＳ１で認識したクライアントの実行形式に適合したＪＩＴコンパイラを選択してＪａｖａプログラムのコンパイルを実行する。

【００５４】図10は、図7のステップＳ４におけるコンパイル装置１８のコンパイル処理の他の実施形態であり、この実施形態にあっては、代理コンパイルサーバ１０に接続可能なクライアントマシンの全ての実行形式に適合したネイティブコードに翻訳してクライアント側に返送するようにしたことを特徴とする。

【００５５】即ち、ステップＳ１で，要求を行ったクライアントとは直接関係せずに予め決められている特定のクライアントの実行形式を設定し、ステップＳ２で、この実行形式に適合したＪＩＴコンパイラを選択して、そのネイティブコードにコンパイルし、ステップＳ３で、コンパイルした実行モジュールをバッファに格納する。

【００５６】続いてステップＳ４で全実行形式のコンパイル処理の終了の有無をチェックし、済んでいなければステップＳ１に戻り、次のクライアントの実行形式を設定し、同様に実行形式に適合したＪＩＴコンパイラを選択してネイティブコードにコンパイルし、ステップＳ３で実行モジュールをバッファに格納する。

【００５７】ステップＳ４で全てのクライアントの実行形式のコンパイル処理が終了すると、そのときバッファに格納している各実行形式に適合したネイティブコードの実行モジュール、即ちコンパイル済みＪａｖａプログラムを要求元のクライアントに返送する。

【００５８】このようにして複数種類の実行形式のネイ

イル済みのコード長が長くなるネイティブコードのＪａｖａプログラムについてはキャッシュ装置１２に保存しないようにすることで、ハードディスクに使用するキャッシュ装置１２としての容量を低減できる。

【００７４】図１４は、図１３のコンパイル前のＪａｖａプログラムのみをキャッシングする場合の図３の代理コンパイルサーバ１０の処理動作のフローチャートである。この実施形態にあっては、ステップＳ１でクライアントからのプログラム要求を受け取ると、ステップＳ２で、要求されたＪａｖａプログラムがキャッシュ装置１２にキャッシングされているか否かチェックする。

【００７５】キャッシングされていなければ、ステップＳ３でネットワーク１８上のウェブサーバ２０にプログラム要求を行ってＪａｖａプログラムをロードし、ステップＳ４でキャッシュ装置１２にＪａｖａプログラムを保存するキャッシングを行った後に、ステップＳ５でコンパイル装置２８によりＪａｖａプログラムを要求元のクライアントの実行形式の適合したネイティブコードのＪａｖａプログラムにコンパイルし、ステップＳ６で、コンパイル済みのＪａｖａプログラムをクライアントに返送する。

【００７６】このコンパイル済みＪａｖａプログラムをキャッシングしない実施例にあっても、コンパイル前のＪａｖａプログラムをキャッシングしておくことで、クライアント側の要求に対し、キャッシングしているＪａｖａプログラムについてはネットワーク１８上のウェブサーバ２０に要求することなく直ちにコンパイル装置２８でネイティブコードにコンパイルして要求元のクライアントに応答することができ、その都度、コンパイルを行う分、図７の処理に比べ時間はかかるが、キャッシュ容量を少なくできる分だけ、代理コンパイルサーバ１０のハードウェア構成を簡略化できる。

【００７７】図１５は、図１３のコンパイル前のＪａｖａプログラムのみをキャッシングする場合について、図１１と同様に、クライアントからの要求に対しプログラムをコンパイルして返送した際に、アプレット４０に対するクライアントファイル４２－１～４２－４のダイナミックリンク４５を解析して、次にクライアントが要求するＪａｖａプログラムを予測してコンパイルする処理を適用したフローチャートである。

【００７８】この図１５の処理にあっても、ステップＳ１～Ｓ６は図１４と同じであるが、クライアント要求プログラムをコンパイルして返送すると、ステップＳ７で図１のようなダイナミックリンク４５の解析に基づき、次にクライアントが要求するプログラムを予測し、ステップＳ８で予測プログラムをキャッシングしているか否かチェックし、キャッシングしていなければ、ステップＳ９でネットワーク１８上のウェブサーバ２０から予測プログラムをロードし、ステップＳ１０でキャッシュ装置１２にキャッシングする。

【００７９】ステップＳ８で予測プログラムがキャッシングされていれば、ステップＳ９，Ｓ１０の処理はスキップする。この場合にも、クライアント側からあるＪａｖａプログラムが要求されてコンパイル後に返送されると、次に要求されるＪａｖａプログラムを予測して予めキャッシングしておくことで、実際に予測が当たった場合にはクライアント要求に対し代理コンパイルサーバ１０より直ちに要求プログラムをコンパイルして応答することができ、ネットワーク上のウェブサーバ２０のアクセス分だけＪａｖａプログラムの実行時間を短縮することができる。

【００８０】尚、上記の実施形態は、代理コンパイルサーバ１０に設けたコンパイル装置２８としてＪＩＴコンパイラを例にとるものであったが、Ｊａｖａプログラムをクラス単位で翻訳する高速バイトコードコンパイラを設けてもよい。高速バイトコードコンパイラは、クライアント側がいつも同じマシンで動作するスタンドアローンアプリケーションの場合に適している。

【００８１】またコンパイル装置１０にメソッド単位での翻訳と最適化を行う機能修正、更新が容易なアプリケーションに適合したＪＩＴコンパイラと、機能が固定されたアプリケーションに適した高速バイトコードコンパイラの両方を設け、クライアント側のアプリケーション環境に応じて選択するようにしてもよい。

【００８２】また上記の実施形態は、ネットワーク上の仮想マシンコンピュータプログラムとしてＪａｖａプログラムを例にとるものであったが、ネットワーク上のプログラムをコンパイルしてクライアント側で実行する形態であれば適宜のシステムにそのまま適用できる。

【００８３】更に本発明は、コンピュータネットワークに設けた代理コンパイルサーバ１０そのものを対象としており、具体的には図１６のようなプロキシサーバにＪＩＴコンパイラの機能を持たせることで実現できる。

【００８４】

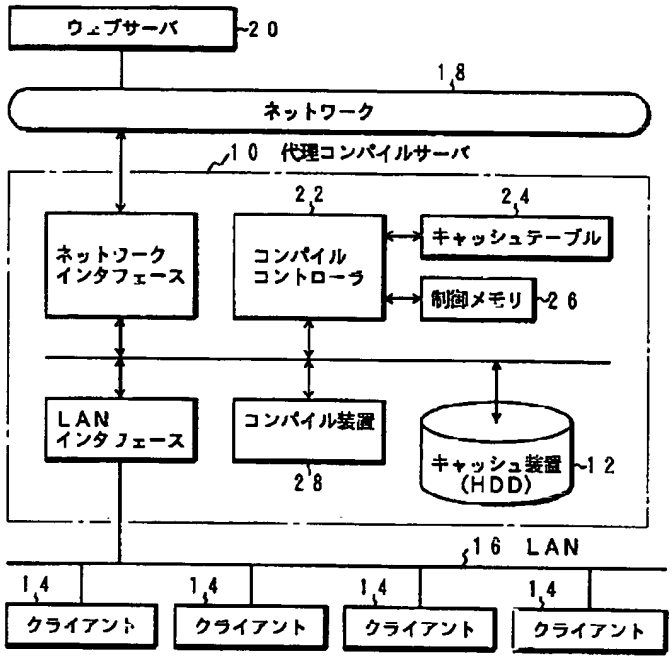
【発明の効果】以上説明してきたように本発明によれば、ネットワークの代理サーバに仮想マシンコンピュータプログラムのコンパイル機能を付加して代理コンパイルサーバとしたことで、ネットワーク上の仮想マシンコンピュータプログラムを実行するクライアントの負担を減らし、ネットワークコンピュータシステムの下でクライアントがアプリケーションプログラムを高速実行することができる。

【００８５】また代理コンパイルサーバにキャッシング機能を設け、クライアントの要求に対しキャッシングしているネイティブコードにコンパイル済みの仮想マシンコンピュータプログラムを返送することで、クライアントからの要求に対するコンパイル済みプログラムの返送が短縮でき、ネットワーク上のプログラム要求から実行までの時間が短縮され、結果としてクライアントにおけるアプリケーションプログラムの実行を更に高速化でき

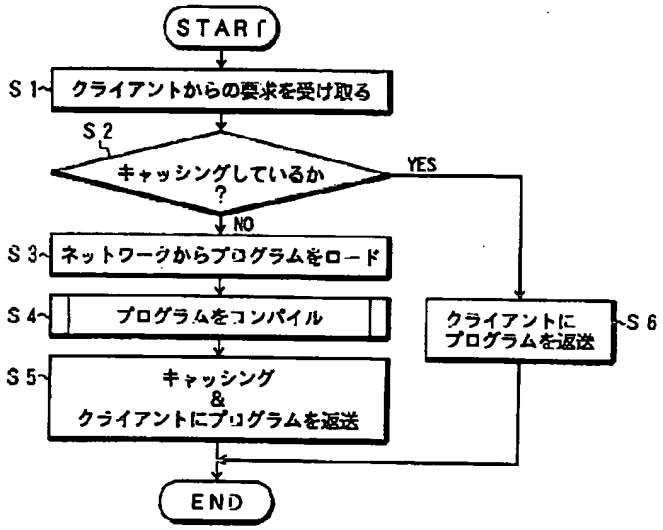
【図1】

本発明の原理説明図

(A)

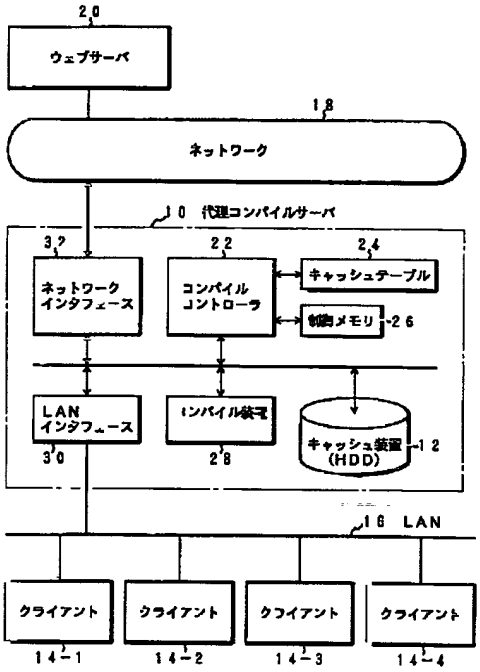


(B)



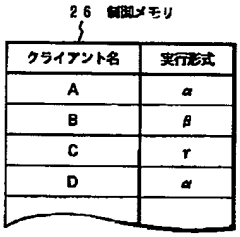
【図3】

本発明のシステム機能のブロック図



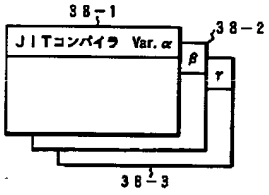
【図5】

図3の代理コンパイルサーバに設けた制御メモリの説明図



【図6】

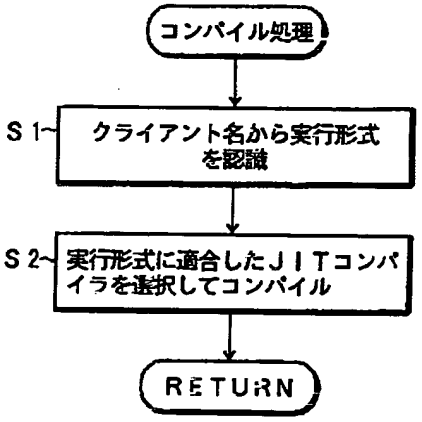
図3の代理コンパイルサーバに設けたコンパイル装置の説明図



【図8】

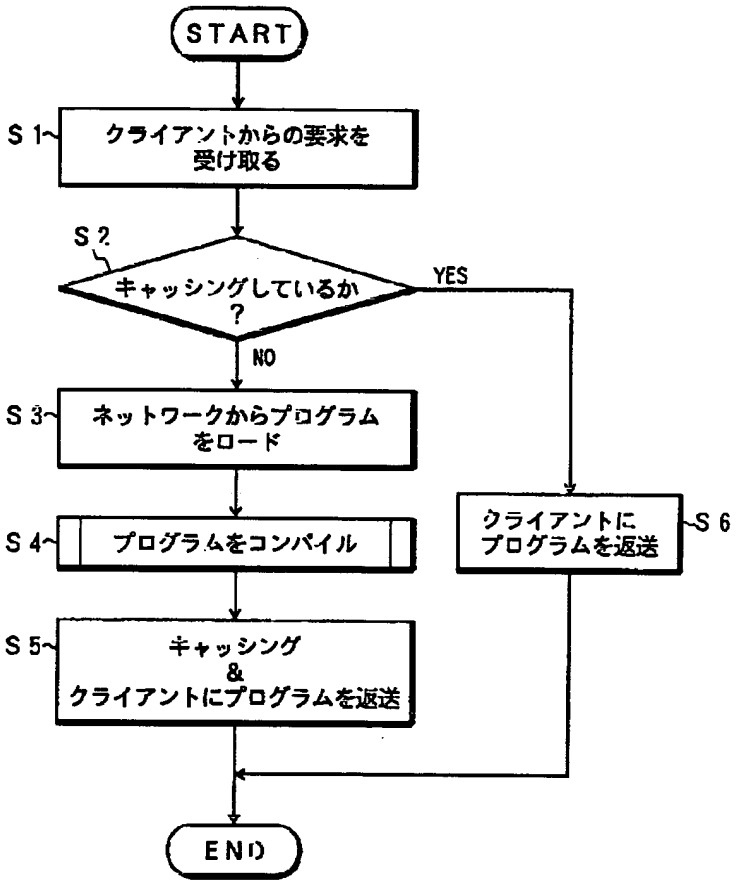
クライアント名から実行形式を認識して行う

図1のコンパイル処理のフローチャート



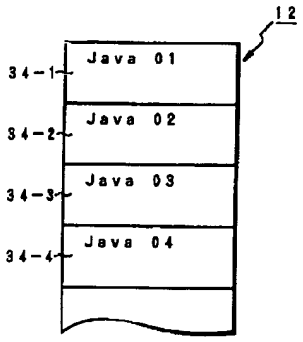
【図7】

図3の代理コンパイルサーバの処理動作のフローチャート



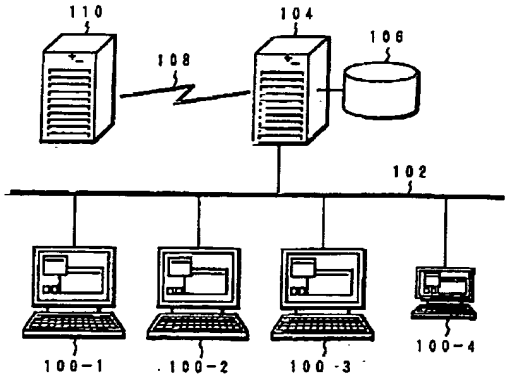
【図13】

ネットワークから受けたプログラムをそのまま保存する図3の代理コンパイルサーバに設けたキャッシュの説明図



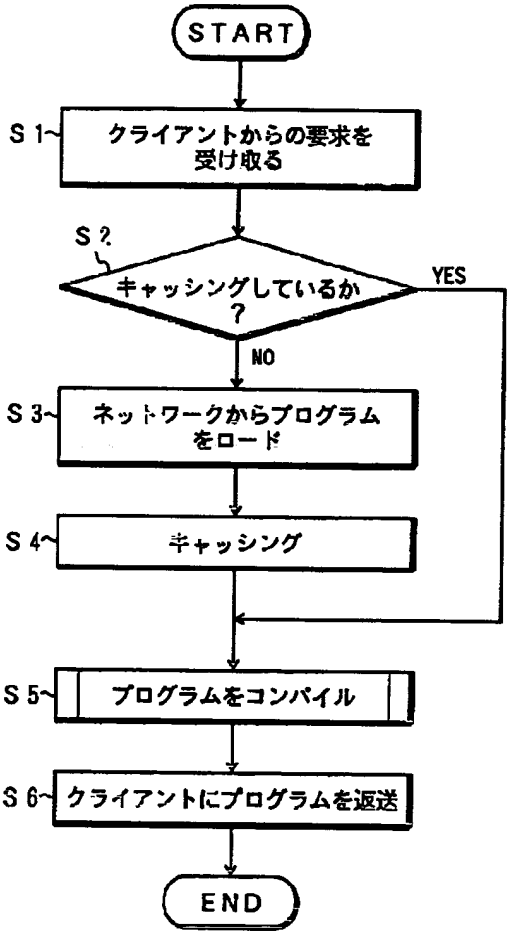
【図16】

従来システムの説明図



【図14】

図13のキャッシュを用いた図3の代理コンパイルサーバの処理動作のフローチャート



【図15】

クライアントの要求プログラムを予測して図13のキャッシュに保存する

図3の代理コンパイルサーバの処理動作のフローチャート